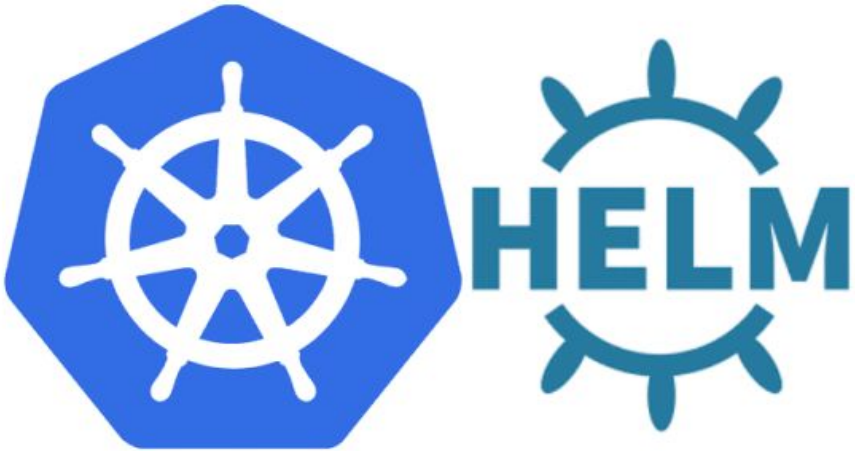
Kubernetes For DevOps

Lev Epshtein


HELM

Kubernetes Helm

Helm is a tool for managing Kubernetes packages called *charts*. Helm can do the following:

- Create new charts from scratch
- Package charts into chart archive (tgz) files
- Interact with chart repositories where charts are stored
- Install and uninstall charts into an existing Kubernetes cluster
- Manage the release cycle of charts that have been installed with Helm

Three big concepts

A *Chart* is a Helm package. It contains all of the resource definitions necessary to run an application, tool, or service inside of a Kubernetes cluster. Think of it like the Kubernetes equivalent of a Homebrew formula, an Apt dpkg, or a Yum RPM file.

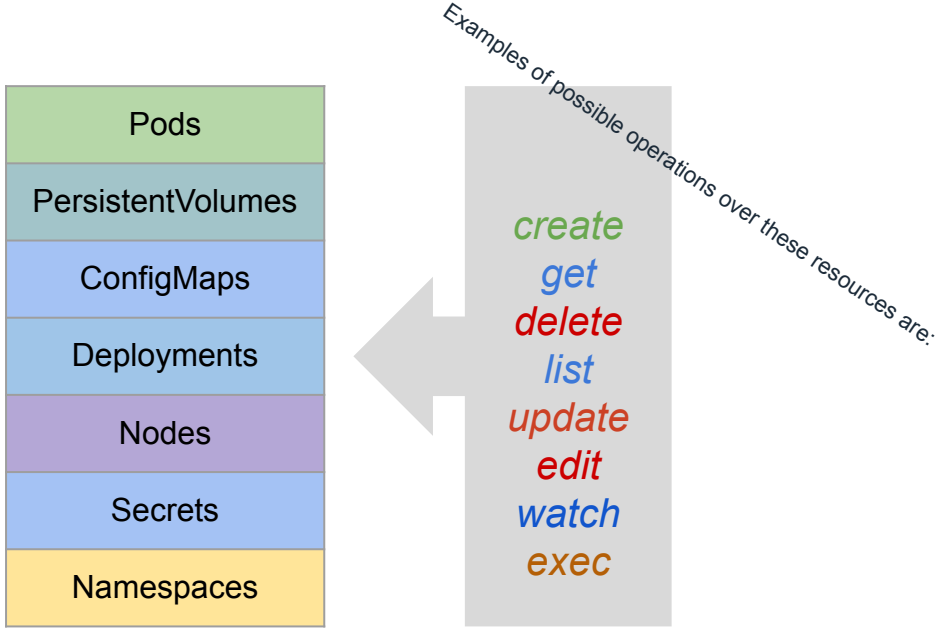
A *Repository* is the place where charts can be collected and shared. It's like Perl's CPAN archive or the Fedora Package Database, but for Kubernetes packages.

A *Release* is an instance of a chart running in a Kubernetes cluster. One chart can often be installed many times into the same cluster. And each time it is installed, a new *release* is created. Consider a MySQL chart. If you want two databases running in your cluster, you can install that chart twice. Each one will have its own *release*, which will in turn have its own *release name*.

RBAC API Objects

One basic Kubernetes feature is that all its resources are modeled API objects, which allow CRUD (Create, Read, Update, Delete) operations.

Examples of resources are:



At a higher level, resources are associated with API Groups (for example, Pods belong to the *core* API group whereas Deployments belong to the *apps* API group). For more information about all available resources, operations, and API groups, check the Official Kubernetes API Reference.

To manage RBAC in Kubernetes, apart from resources and operations, we need the following elements:

- Rules
- Roles and ClusterRoles
- Subjects
- RoleBindigs

Follow tutorial:
https://docs.bitnami.com/kubernetes/how-to/configure-rbac-in-your-kubernetes-cluster/